DOCUMENT RESUME

ED 355 773 FL 021 026

AUTHOR	Mao, Tang
TITLE	Courseware Authoring and Delivering System for Chinese Language Instruction. Final Report.
INSTITUTION	Comptek Co., Springfield, NJ.
SPONS AGENCY	Department of Education, Washington, DC.
PUB DATE	5 Feb 85
CONTRACT	300-84-0180
NOTE	41p.
PUB TYPE	Reports - Descriptive (141)
EDRS PRICE	MF01/PC02 Plus Postage.
DESCRIPTORS	*Authoring Aids (Programing); Chinese; *Computer Assisted Instruction; Computer Software Development; *Courseware; *Programing; Research Projects; Second Language Instruction; *Second Languages; *Uncommonly Taught Languages
IDENTIFIERS	*Unix Operating System

ABSTRACT

A study investigated technical methods for simplifying and improving the creation of software for teaching uncommonly taught languages such as Chinese. Research consisted of assessment of existing authoring systems, domestic and overseas, available hardware, peripherals, and software packages that could be integrated into this project. Then some features of an authoring system were applied to a personal computer using available hardware and software. Among the findings were that interactive computer-assisted instruction (CAI) applications have large storage requirements; choice of hardware and peripherals require trade-offs in cost, speed, and adequacy; a courseware author is needed; the program can be designed to incorporate a third language; data compatibility is the primary difficulty in transferring information from one computer type to another; and the Unix system is inadequate for this application. It is concluded that an interactive CAI system for multiple languages with graphics, text, and voice capability at reasonable cost can be developed if special care is given to the needs of potential users, system development, and integration of hardware and software. The final product of the research is a detailed design for a multi-language, multimedia courseware authoring and delivery system. (MSE)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED355773

ERIC

Department of Education
Contract No. 300-84-0180

FINAL REPORT COVER SHEET

February 5, 1985

Small Business Innovation Research Program Phase-I

Topic 2: Simplifying and Improving the Creation of Software

Project Title: A Courseware Authoring and Delivering System for Chinese Language Instruction

Prepared By: COMPTEK Company
P. O. Box 245
Springfield, N.J. 07081

ACKNOWLEDGEMENT

This material is based upon work supported by the U.S. Department of Education under contract number 300-84-0180. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views or policies of the Department of Education.

ENDORSEMENTS

Principle Investigator
Name: Tang Mao
Title: Manager, Government Contract
Signature Tang Mao
Date 2/25/85
Phone No. (201) 5648711

Business Official
Name: Vincent M. Lai
Title: Vice President
Signature Vincent Lai
Date 2/26/85
Phone No. (201) 5648711

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☒ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

FL021026

BEST COPY AVAILABLE

FINAL REPORT SUMMARY

February 5, 1985

Prepared By: COMPTEK Company,
P.O. Box 245, Springfield, N.J. 07081
Tang Mao, Manager, Government Contract

Department of Education, RFP No. 84-022, Contract No. 300-84-0180

Topic 2: Simplifying and Improving the Creation of Software
Project Title: A Courseware Authoring and Delivering System for Chinese Language Instruction

Technical Abstracts and Results

This research studied the feasibility of a user-friendly and cost-effective courseware authoring and delivering system for Chinese language instructions. Among other areas, discoveries were made in

-- Modular design for multiple foreign language CAI systems.
-- Storage requirements of interactive CAI systems.
-- Human interface for courseware authors.
-- Difficulties and certain solutions about transferring courseware among heterogeneous systems.

The conclusion is that an interactive CAI system for multiple languages with graphics, text, and voice capabilities at a reasonable cost can be developed, provided special care and judgement is given to the need of potential users, the system development, and the integration of hardware and software. The final product of this research is a detailed design of a Multi-language, Multi-media, courseware Authoring and Delivering system (MMAS).

Key Words:

Courseware, Portability, Chinese, CAI, Authoring System, Language Instruction.

Contributions and Potential Applications:

The results of the research would have contributions on:
-- Man-computer interface
-- Software engineering
-- Integration of advanced hardware and software techniques for interactive CAI systems.

Table of Contents

1. Executive Summary 1-1

2. Research Findings 2-1

3. Design of MMAS Authoring System 3-1

4. Design of MMAS Delivering System 4-1

5. Design Discussion 5-1

6. Related Work and Existing Products 6-1

7. Bibliographies 7-1

Appendices

A. User Interface Specification for MMAS Authoring System A-1

B. Database Structure for MMAS Authoring System B-1

C. Glossary C-1

1. Executive Summary

The purpose of this project is to research into the feasibility of a user-friendly and cost-effective courseware authoring and delivering system that can teach uncommonly taught languages such as Chinese. *

We conducted studies about existing authoring systems, both domestic and abroad, available hardware, peripherals, and software packages that can be integrated as parts of our product. Their advantages and shortcomings were identified. We have also interacted with many potential users of the system and professional educators. The many suggestions and feedbacks were highly valuable in designing the system.

To further examine the feasibility of such a system, we implemented some features of an authoring system on an IBM-PC XT using available hardware and software. The purpose of the implementation is to study the requirement of the resources for the system, and to explore the performance of certain peripherals and software packages that are candidates for the system.

Some research discoveries in Phase-I are:

1. The large storage requirement of interactive CAI applications.

2. The trade-offs of various hardware and peripherals in terms of their costs, speed, and adequacy in incorporating into our system.

3. The recognition of the need of a courseware author.

4. The feasibility of modular designing an interactive CAI system to teach not only Chinese, but other non-Romanic languages.

5. The difficulties and certain solutions about transferring courseware among heterogeneous computer systems.

---

* Parts of this report were included in the Phase-II proposal, which was submitted to U. S. Department of Education under Solicitation Number NIE-R-85-0005 in January 1985.

6. Current inadequacy of using Unix in the interactive CAI applications.

The conclusion of the phase-1 research is that an interactive CAI system for multiple languages with graphic, text, and voice capabilities at a reasonable cost can be developed, provided special care and judgement is given to the need of potential users, the system development, and the integration of hardware and software. The final product of this phase of the project is a detailed design of a Multi-language, Multi-media (graphics, text, and voice), Authoring and Student system (henceforth referred to as the MMAS system.)

The results of the Phase-I research would have contributions on:

- Man-computer interface,
- Software engineering,
- Database conversion and information porting among heterogeneous computers, and
- Integration of new hardware and software techniques for the development of interactive CAI systems, which we believe to be the main contribution.

In the rest of the report, Section 2 will discuss each of the discoveries in detail. Sections 3 and 4 will describe the design of the MMAS systems. Section 3 is devoted for the authoring system, and Section 4 is for the delivering system. Section 5 contains discussion about the design and an extension of the system to Japanese instructions. Finally, Sections 6 and 7 list related work, products, and bibliographies.

2. Research Findings

2.1 Storage Requirements for interactive CAI Applications

An important finding in Phase-I is about the storage requirement for an interactive CAI applications. We did experiments with two standard storage methods proposed for graphic and voice data: VDM and NAPLPS [Hin84] [Gss84]. The amount of storage required for a still frame depends on the contents of the screen and the compression algorithms used. Our experiments showed that a typical color graph courseware screen takes from 5,000 to 200,000 bytes in the VDM format. It takes more storage in the NAPLPS format.

With these data and the assumption that a student on average takes 20 seconds to study and respond each frame, it can be shown that a student using a low-cost personal computer based CAI system needs to replace a new floppy disk at least every 10 minutes. # For a CAI course of 1 hour length, a student will replace floppy disks for 12 or more times, about the same number of times he (she) raises his hands in a normal class! Although this limitation does not remove low-end personal computers completely from the CAI market, it does severely restrict their potential applications.

To remedy this limitation, there are a few new technologies available. Among them are video disks and optical disks. Their trade-off in terms of performance, cost, and adequacy in the interactive CAI will be examined in the next section.

The figure is derived by the following logics: Given 20 seconds for a frame and 5,000 bytes for a frame, it takes 15 kilobytes for each minute of CAI material. Furthermore, in an interactive course, approximate half or more materials are not played to the student. This can be illustrated with an example. For each question the interactive courseware needs to have materials for both correct and incorrect answers. Students answer the question correctly would not see the part of courseware that corresponds to the incorrect answer. This amounts to 30K bytes for every minute of courseware. Virtually all low-cost personal computers use floppy disks for mass storage, which mostly have capacities ranging from 200 to 400 kilobytes. This low capability implies that the courseware is exhausted in 10 minute or less.

2.2 Trade-offs Among Hardware and Peripherals

Two peripherals are important to an interactive CAI system: the storage device and the voice device. The storage device contains the courseware and the voice device provides voice for the course.

2.2.0.1 Storage Devices [Rot84] [Sohu84] [Mol84a] Besides the floppy disk discussed in the last section, there are many other storage devices. Among them are: video casstte recorder, video disk, hard disk, and optical disk. Their adequacy as the storage device of the interactive CAI system is the subject of this section.

Video cassettee recorders (VCRs) are popular home entertainment equipments. They are inadequate in an interactive CAI environment however, because their nature of "sequential access." Any one who has used a VCR realizes that it takes a few minutes to completely rewind a VCR cassettee. Moving from one point on the VCR tape to another point takes many seconds to even minutes. In an interactive CAI environment where response time to the student is critical, the slow access time disqualifies VCRs as a candidate for the application.

In comparison, disks have much faster access time. Typical access time for any data item on a disk is from 10 to 500 milliseconds (One millisecond is one thousandth of a second.) This speed is adequate for the interactive CAI application. The three types of disks will be discussed in the following:

Video disks are popular home entertainment equipments. They are considered less appealing in comparison with the other two disks for the two reasons: First, the video disk contains analog information, but computers process only digital information. Converting between these two forms of information takes special hardware and efforts. Second, currently all video disk players are designed for non-interactive environment. Moving disk header frequently, as is necessary in an interactive CAI application, requires special hardware.

Optical disks (also called laser discs) stores information with laser mechanism, compared with magnetic media used in video disks and magnetic disks. They are used to store digital information and are appropriate for the interactive CAI application. Another advantage of optical disks is their storage capacity. Optical disks have the largest storage capacity in all computer storage devices. Depending

on models, their capacities range from 50 megabytes to a few hundred gegabytes, * which are more than sufficient for CAI applications.

One shortcoming about optical disks, however, is that so far all optical disks are "write-once" only. Once any information is stored on the disk surface, it cannot be over-written. Therefore the optical disk is inadequate for courseware development, such as the MMAS authoring system, since the courseware need be edited and modified many times before being finalized. As a matter of fact, even the write-once disk is too expensive and has a too large a physical dimension for use in personal computer applications. The latest technology on optical disks is the "read-only" disks. Disks are produced in a central factory in large volume for distribution, just like music record albums. There are a few firms specializing in producing pre-recorded optical disks [Pio82] [Pio83]. The cost for each copy is low, yet the making of the master copy costs from $50,000 to $200,000, still too expensive for any wide applications.

Read-write optical disks are being developed now, but are not expected for another two years. Mass production may take even longer.

The third kind of disks are magnetic disks. These disks can be written and read for as many times as one wants. They are adequate for courseware development applications.

Magnetic disks can be categorized as removable and non-removable, depending on whether the disk pack can be removed physically from the computer. The removable disks manufactured for the microcomputers are called floppy disks. The non-removable disks are called hard (or Winchester) disks. The shortcoming with floppy disks is their relative small storage capacity, which was discussed in the last section. Hard disks have larger storage capacities, ranging from 5 to hundreds of megabytes. This capacity is sufficient for most courseware developments. A disadvantage of hard disks for microcomputers is, however, their non-removable nature. The disk packs cannot be simply moved, like optical disks or floppy disks, from one system to another. After the development of a courseware is

---

* One gegabyte is a billion bytes, and one megabyte is a million bytes.

completed, special arrangements are necessary to move the courseware to delivering systems.

With all the requirements and limitations described above, designing an interactive CAI system takes special care and conscious selections.

2.2.0.2 Voice Devices

Voice devices are necessary in CAI systems that tutors foreign languages to provide students with a means to hear and imitate the voice.

A popular approach to provide voice capability is the voice synthesizer. A voice synthesizer takes alphanumeric words and pronounces voices from a speaker. It was among the potential candidates we researched in the phase-I project. After examining products by Texas Instruments, Xerox, and several other firms, we decided voice synthesizer is inadequate for our application for the following reasons: First, voice synthesis technique has not been applied to languages other than English and Japanese. Although not very difficult in our assessment, a good Chinese voice synthesizer will not become available for about 3 years. Second and the ultimate reason is that we simply believe voice synthesizer is not the way to teach language lessons. In language lessons that students imitate, not only the language spoken need be recognizable, but also the quality of the voice is very important. Voice synthesis techniques may find their usage in manufacturing or office automation applications where recognition of words is sufficient. Yet anyone has been exposed to a latest voice synthesizer would have realized that it still has a long way to go to speak like a real human being. Put it simply: No one wants to learn to speak like a machine.

The voice peripheral we chose for the MMAS system is the digital voice device [Dia84].

2.3 The Recognition of the Need of the Courseware Author

During Phase-I, we interacted with many educators, students, and authors who are interested in developing or learning courseware on the MMAS system. One important finding was that an author is an author, not a cartoonist. He (she) is often more interested in preparing a course effectively than playing with fancy graphics or colors. This recognition leads to two goals of the human interface for the MMAS system:

- User friendly, and

- Effective

In the design of the MMAS authoring system, much effort was devoted to make the author interface friendly. The MMAS provides the menu feature to direct the author, so at any time, entering one common command displays for the author the current state of the courseware being developed and the options he (she) may choose.

To make the author effective in composing courseware, we provided the library of frames, the window feature [Gla84], and the editing capabilities for graphs, texts, and voice messages. About the library function, not only the MMAS system provides certain re-usable graphic frames and tokens, but also the author can create and keep such a library. For instance, suppose an author finds one graphic frame to be useful in the presentation of the courseware, she (he) can simply save that frame as a library object and retrieve to use in the future courseware development.

A more complete description about the author interface of the MMAS system can be found in Appendix A.

2.4 Feasibility of Incorporating a Third Language in MMAS

It was discovered that the MMAS can be designed in such a way that it can be converted, with minor effort compared with the development of a new system, to teach language instructions of a third language such as Japanese. The key nature to be kept in the whole design is the principle of modular design [Boe78] [IEE83].

This aspect will be further discussed in Section 5. After the design of the MMAS system is presented in Sections 3 and 4. it would be easier to explain.

2.5 Tranferring Courseware among Heterogeneous Computers

When information is transferred from one type of computers to another type of computers, data compatibility is the main obstacle. Since courseware is a particular type of data, porting courseware between heterogeneous systems has similar problems. Our research in Phase-I discovered that there are two levels of courseware compatibility problem.

1. Storage hardware and data format

   All CAI systems need certain mass storage for the courseware. Moving physical storage hardware such as

disks between systems often encounter difficulties. Illustrate with an example: most microcomputers have floppy disk interface, yet not all floppies are compatible with each other. One instance of hardware incompatibility problems happens when one computer uses floppies of 5 and 1/4 inch size, yet another uses only 3 and 1/2 inch size. Even for the same size floppies, recording density and format differences often prevent one type of computers from reading disks of another type.

2. Data representation and data structure alignment

   If the first compatibility problem is solved, each byte can be transferred accurately. The second type of incompatibility problem, data representation and data structure alignment, may arise. The problem of data representation is related to the binary representation of data. For instance, in IBM-PC an integer is formed by combining two bytes, the lower-address byte has a higher value. In a DEC PDP-11, the order of the two bytes is reversed. So for the same byte contents, they mean one data value to one computer, but a complete different value for another computer. The data structure alignment problem is related to both computer hardware and computer language compiler. For certain computers, a long integer (need four byte storage) must start at addresses ended with (octal) 0 or 4. But for certain other computers, a long integer may start at double byte addresses (i.e. address values ended with octal 0, 2, 4 or 6). Since compilers take this advantage and move the alignment of data structures to reduce the code generated, yet other compilers don't, all these pose difficulties when courseware is to be transferred to a complete different computer.

For the first type of hardware problems, one solution is getting compatible hardware. For instance, to generate floppy for Apple McIntosh computers, we'll find a floppy drive that can generate 3 and 1/2 inch floppies.

For the second type of problems, there are a few solutions:

1. The authoring system generates only one kind of standard format. Each delivering system has utilities to translate the standard format into its native format.

2. The authoring system generates one type of format for each different system. The translation utility

software will reside on the authoring system.

The MMAS will use the second approach. The reason is that the authoring system has more resources, both in terms of CPU and storage space, than the student system. Also, in this way the students need not to learn how to use the utility software. They only need to get the right type of floppy or optical disks, simply insert into their delivering systems, and start to learn.

2.6 Current Inadequacy of Using Unix in the CAI Applications

In our proposal for Phase-I, we had indicated the attractiveness of developing the authoring system based on Unix operating system. It is a newer generation operating systems than most systems on mainframe computers, and is gaining popularity in the mini-computer community.

During Phase-I, we did an in-depth analysis on Unix. The strength of Unix is mainly in the software development area. It is well known for its excellent software tools such as screen editors and compilers, and is gaining its market shares in the business computer applications. The influence of Unix has not, however, reached the CAI area. There exist few advanced voice peripherals and few graphics packages like GKS and VDI [GKS84] [Mol84] [Mar83] [Raw83] [Wag84] [Hin84], which are used as cornerstones of the MMAS. In comparison, DOS operating system, supported by computer giant IBM, and Apple's proprietary operating system have a dominate presentation in the CAI community. Many advanced CAI peripherals and graphics packages are available for these operating systems. We have therefore decided to use instead DOS operating system.

3. The Design of the MMAS Authoring System

This section presents the design of the authoring system for the MMAS system. It will be stated in the order of hardware and software. The hardware section consists of the base computer and the peripherals, and the software section consists of both software that will be purchased from outside and software that will be developed in house. Functions of each software module will be discussed. First time readers may wish to skip the next two Sections and proceed with Section 5. The materials in these two Sections may be reviewed later.

3.1 Hardware Architecture

The computer of the authoring system will be the IBM Personal Computer (PC) XT model. It has an Intel 8088 CPU, a 350 kilobyte floppy disk, and a hard disk of 10 megabyte capacity.

Special peripherals include the IBM graphics controller, keyboard, and the voice device from Dialogic Company. The graphics controller is connected to a color monitor. The keyboard is the input device, from which the author inputs both English commands, texts, and Chinese characters. The voice device includes a computer board to be installed inside the PC, a speaker and a microphone. The voice device, along with its driver software, processes the voice input from the microphone, compresses and stores as files on the disk. Under software control, the voice can be assembled from those voice files and re-produced at the speaker. The system software provides voice editing capability.

The operating system will be DOS 2.0, which is the most popular operating system for the IBM-PC family.

3.2 Software Architecture

The software architecture for the authoring system is illustrated in Figure 3-1. In the figure, each block represents a logic unit, or module, of software. The size of a module in the figure has nothing to do with its real program size. Modules are grouped into four layers, each contains a few modules. The convention used here is that a module at a higher layer may invoke services from one or more modules at its lower layer. The numbers appearing in parentheses (e.g. (3)) are used only for referencing to a module when mentioned in the context. Furthermore, the purpose of the figure is to illustrate the architecture of the MMAS, consequently not all modules in the system are

shown. Modules not shown in the figure are considered insignificant in this context. As an example, many modules in the DOS operating system are not shown in the figure.

The lowest layer of modules consists of device drivers which are part of the operating system kernel. These drivers directly interface with peripherals such as the disk, the monitor, and the voice device.

The immediate layer above the operating system is characterized by the common nature that they are existing software, either can be purchased from software vendors or comes with a particular peripheral. The former includes the GKS package (8), the window routines (9), the VDM package (10) and the foreign database subroutines (11). The latter includes the voice subroutines (7), which came with the vendor of Dialog/I voice peripheral.

The next two higher layers consist of modules that will be developed in house: Voice Editor (2), Frame Editor (3), Rehearser (4), Flow Organizer (5), MMAS Database (6), and Menu-Driven Controller (1). Voice Editor facilitates the editing of voice files; Frame Editor enables the editing of monitor screen, which includes graphics, English and Chinese text in various fonts; Rehearser allows the author to replay the courseware being prepared; Flow Organizer allows the author to organize the material of the courseware in a structured and automatic way; MMAS Database provides access to graphical frames, courseware organization, and Chinese character database. Finally, the Menu-Driven Controller provides an overall control for the authoring system. The three modules Rehearser, Frame Editor, and the Flow Organizer interface with the courseware author via the keyboard and the monitor, and will be the subject in Appendix A, Author Interfaces. The Database will be discussed in the next section.

3.3 Database and Courseware Structure

This section describes the structure of the database and the courseware, which is being developed in the MMAS authoring system.

The database in the authoring system consists of mainly libraries of graphics, foreign language characters, author's own frame library, etc., which may be retrieved by the author to create the courseware.

We shall devote more space to the structure of a courseware. A courseware is represented as seven types of data files, each references to one another. Their names are listed

below and their functions are explained in the following paragraphs:

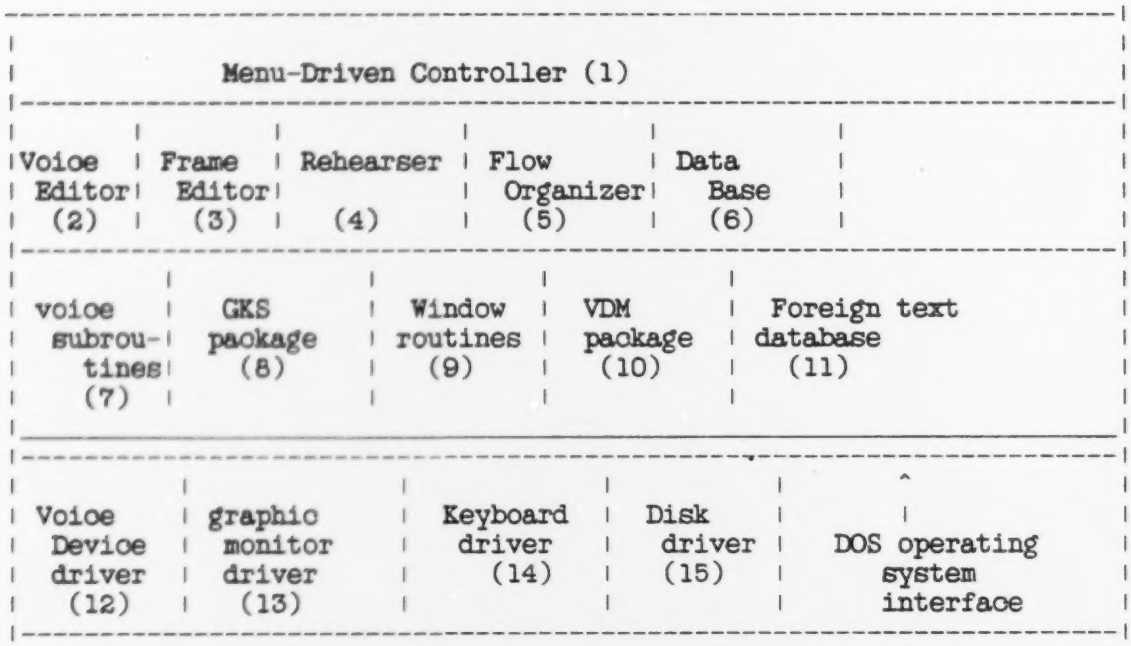


Figure 3-1 Authoring System Software Architecture

1. Node-file
2. Graph_file
3. Text_file
4. Voice_files
5. Special_Graph_file
6. Control_command_file
7. Master_file

3.3.0.1 Node File This kind of files contains information that corresponds to the "scene" or the "update" as described in Appendix A, User Interface. Each "scene" is represented

as a record in the file. Each record consists, among other information, indicators to other files, namely Graph_files, Text_file, Voice_file, and Control_command_file. For a more detailed definition of the node file, the reader is referred to Appendix B, where a complete definition of the structure for the node is presented.

3.3.0.2 Graph File This kind of files contains information that describes a screen graph. The graph is recorded with the VDM format. The standard enables the use of other standard-conforming software in a student system to display the graph.

3.3.0.3 Text File This kind of files contains only English or Chinese character text and their screen coordinates to be displayed. The VDM format has provisions to represent English text, and represent graphical items, which can be used to express Chinese characters. Theoretically we can save all information in the Graph-file and do without the Text-file. The reason we choose to provide this text file is two-folded: First, much storage space can be saved by storing Chinese character separately in another data file, and be referred by an index to that file (This is called indexing in computer jargon.) Second, some text may be used over and over in many frames in the same course. For example, the sentence "To hear the voice of the word, hit the RETURN key" may be used by most frames in a languge course. Saving it in a separate record enables the repetitive usage straightforward. Finally, if only text data are to be placed on the screen, it improves performance by calling a text input/output routine than calling a general purpose graphic interpreter.

3.3.0.4 Voice Files This kind of files store information about voice information. Since contents of these files are used to feed voice hardware, the format of these files is incomprehensible by the software.

The Dialog voice device of the MMAS system requires each sentence be saved as a separate file. There will, therefore, have as many files as the sentences in a course. Since the data structure as described in Appendix B has only one fixed size field for the voice file, all voice files will have the name VOxxxxx. When the field has a value, say 13629, it refers to the voice file named VO13629.

3.3.0.5 Special Graphic File Like the graphic file, the special-graphic-file contains information about the graphic tokens, except the information is all about the foreign languge. For a Chinese MMAS system, this file contains all Chinese characters in the particular course.

Note there is no need to put all 3,000 or so Chinese characters in this file. Only one library of all characters is needed in the authoring system. For each course, only those used need be in this file.

3.3.0.6 Control Command File This is the file that saves all control actions that are created by the author. They have the format of character strings, each represents an action as specified by the author.

3.3.0.7 Master File Each type of files may have more than one instance of files, depending on the length of the course. For instance, in the DOS operating system, the maximum length of a file is 64 thousand bytes. When a course's material extends beyond that limit, another file of the same type must be used. The number of files, their relationship with one another, and

4. The Design of the MMAS Delivering System

The delivering system is essentially a slim-down version of the authoring system. The difference with the authoring system is that the delivering system need only be able to interpret the courseware, but need not modify the contents of the courseware.

4.1 Hardware

The hardware required is the IBM-PC with color controller and the monitor. The Dialogic voice device will be installed to provide voice interface. Two variations of mass storage devices may be provided: the two floppy disk drives, or floppy disks with an optical disk drive.

4.2 Software Architecture

The design of the software architecture is illustrated in Figure 4-1.

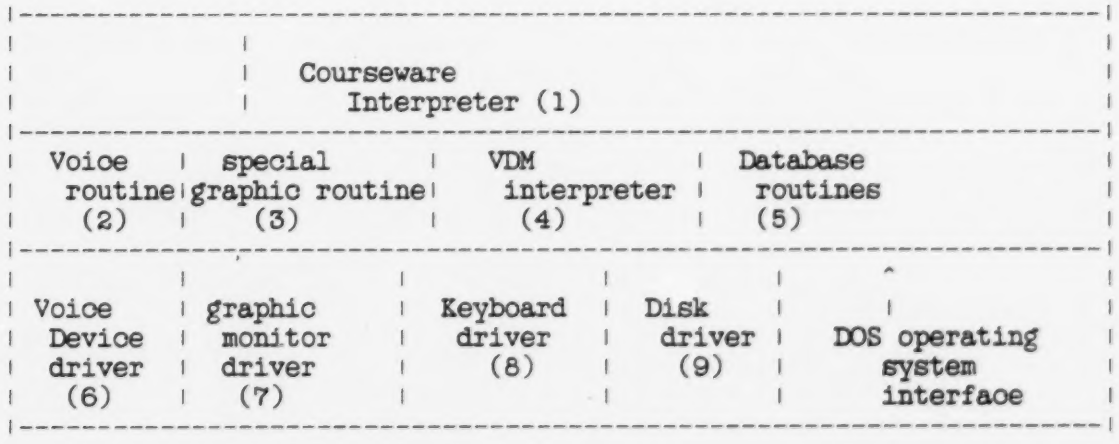


Figure 4-1 Software Architecture of the MMAS Delivering System

The software can be divided as two parts: the system part and the application part. The system part is the software that is contained in the operating system (DOS). It consists of drivers for the graphical controller, the voice device, the disk, and the keyboard. The application part is mainly

the interpreter, which reads the course material from the mass storage, the floppy disk or the optical disk, and outputs the material to the monitor or the speaker. It also senses inputs from the student and reacts according to the instruction stated in the course material.

A more detailed description can be provided by discussing each of the modules invoked by the interpreter.

- The voice routine (2) reads the voice data files, calls the voice driver in the operating system, and produces voice at the speaker.

- The VDM graphic interpreter (4) reads the Graphic_file and Text_file, calls the graphic driver and produces graphs and text at the color monitor.

- The special graphic interpreter (3) reads the Special_Graphic_file and outputs to the color monitor. In the MMAS, the special graphic file contains the Chinese characters.

Note that the special graphic interpreter is itself a general purpose software. It has no knowledge whatsoever about the contents of the database. As long as the database conforms to the format it expects, it simply translates according to the format and outputs the resut. This is the way how "language independence" is reached. With this generality, the same interpreter can be used as student systems for instructing other languages, among which are Japanese, Korean, Russian, and Arabic.*

---

* From CAI application's point of view, Arabic is more similar to English than to Chinese. Arabic consists of 32 basic characters and 3 diacritics, which are equivalent to the 26 English letters [Yaj83]. Unlike the large set of Chinese characters, a small set, such as the one with Arabic is probably more suitable to take a hardware generator approach like the standard ASCII terminal than the software approach taken in MMAS.

5. Design Discussion

This section examines one particular property of the design described in the last sections.

A principle that this project follows closely is the methodology of modular design, [Wir74] [Dij76] [Boe78]. Also referred to as information hiding, this methodology has had important impact on all software developments since early 1970. In a nutshell, the methodology tells that software ought to be grouped into modules, or units of runnable code. A module that conforms to the methodology would provide certain service to other modules yet hide information that is not relevant to the service provided. For example, a monitor driver fits the definition of a module, since it provides ways for other modules to output data to the monitor screen. Yet much of hardware dependent information, such as the interface protocol between the monitor and the system, is "hidden" by the driver, and therefore not known to other modules. The advantage for such a driver is the property of device independence. If and when the monitor need be replaced with another kind of monitor, the only change to the system is the driver software. As long as the new driver provides the same service the original driver does, all other modules need no change at all. This nature provides a stable system architecture, and eases the porting of software.

To further elaborate this desirable nature built in the design of MMAS, we shall discuss one extension to the MMAS: adding a third language, Japanese, to the MMAS.

5.1 Extending the Language Capabilities to Japanese

To study the work that is necessary to modify a Chinese-English based MMAS authoring system to accommodate Japanese, we shall look at the design architecture in Figure 3.1. The only software that need be modified is the foreign text database (11) and possibly the input keyboard device driver (14).

The foreign text database contains the graphical information about characters of the foreign language. The database is used to map from input key strokes to the characters. In the original design, the database contains all Chinese characters that may be used in courseware development. The database is organized as key-pattern pairs. For example, in the standard alphanumeric keyboard input method for Chinese, as discussed in Section 6.6, a sequence of key strokes : Control-C, h, u, l, typed by a user is processed by the menu driven controller (1) by first searching in the foreign text

database. The corresponding Chinese character, matched and retrieved from the database, is a bit-mapped graph. If the graph is output to the screen, for a Tsan-Jae database, it looks like

which is the Chinese word "rock."

Clearly for each language the MMAS system requires a new database that provides a new set of mapping and graphics combinations. To modify the MMAS system to incorporate Japanese, the first step is to replace the foreign text database with a Japanese one.

Japanese has some similarity to Chinese, yet the difference is still large. Japanese consists of two sets of characters: Kana and Kanji. Kana consists of 50 characters, which correspond to the phonetic symbols commonly used in English dictionaries. For most situations, Kana is sufficient for expressing purpose. For situations where ambiguity may arise, especially in the use of nouns, Kanji characters are necessary. Although derived from ancient Chinese, Kanji characters have evolved to differ significantly with modern Chinese characters in the past thousands of years.

One observation we wish to call attention to is about the way the foreign language text is treated in the MMAS. The Chinese (or Japanese) character has no "semantics" to the MMAS system. It treats the character like one of the many graphic tokens, of which one example is the blinking-eye token, in the MMAS library. The MMAS system retrieves and stores a Chinese character just as it does to a graphic token, but does not do anything that is specific to the Chinese character. Using a computer jargon, the Chinese character is treated as a "data" item rather than an "instruction" item. And just due to this reason, it is straightforward to change from Chinese to another language.

Modification of input driver is necessary when another input device, instead of the standard alphanumeric keyboard, is used. As discussed in Section 6.6, both augmented keyboard and big keyboard methods have been used in inputting Chinese and Japanese. In certain applications, these two methods were used frequently. Since these keyboards differ with the

standard keyboard, a new device driver will be needed. The impact to the whole system architecture will be isolated in that driver and the input module, however, and will not affect other parts of the system. For other modules in the system, they get only the same input strings, which are non-distinguishable from those from the original device.

5.2 Compatibility for Voice Data

The principle of modular design benefits a lot to the transferring of courseware between heterogeneous computers as long as there exists a standard that can be followed. For instance, graphic frames are stored in VDM format. As long as another computer understands the same format, (and the structure and alignment problems are solved, as discussed in Section 2.5,) the graphic frames can be displayed on another system.

The important issue is the existence and the conformity of the standard. As an opposite example, consider the digital voice device used in the MMAS system. There does not exist a standard for the voice data format, and the voice files as described in the database of a courseware (Section 3.3) are understood only by that particular hardware. Without common hardware and software that cushions the hardware differences, the voice part of the courseware simply is not portable.

We hope that in the future Dialogic Company will produce voice hardware that can run on different computers yet understand the same format of voice data. For the time being, to port voice courseware to a heterogeneous computer, we plan to construct an electronic device to interface at the electric current level. The hardware will take the electric current that drives the speaker of the MMAS authoring system, bridges to the voice device on the other computer as input, and creates new voice data on the new machine.

6. RELATED WORK AND EXISTING PRODUCTS

The majority of research work in CAI appears to be concentrated in four areas: the United State, the United Kindom, Canada and Japan [Cha80] [Rus83]. As of December 1984, many CAI products exist or have been announced by vendors. In this section, we shall identify some CAI activities in these countries categorized by the related issues of courseware design. In the latter part of this section, existing CAI products are studied.

6.1 Courseware Design Languages

There are two categories of courseware design languages. The first category contains computer programming languages, such as FORTRAN, APL, PASCAL, SMALLTALK, LISP [Spe78], BASIC [Cal79] etc. CONDUIT courseware, which supports higher education classes, is written in BASIC or FORTRAN [CON79]. Bork at UC Irvine adopts APL as a language for interactive graphics to produce courseware of physics instruction [Bor81].

In authoring systems using this category of languages, courseware production requires programmers in a team or the courseware authors have to learn computer programming languages. Also since the above programming languages are designed for some general applications other than developing courseware, they have some features CAI does not need and miss some features that CAI needs. Although these missing features can be implemented as procedures in the programming language, the system performance may degrade.

The second category contains dedicated courseware design languages, such as CAN, PILOT, Coursewriter, ASET and TUTOR [Avn76]. TUTOR, which is derived from FORTRAN, is used by PLATO [CDC83]. Most of the above languages are machine dependent except CAN.

The courseware languages in this category provide higher level tools for the courseware authors. For example, TUTOR has commands for presenting, calculating, data keeping, judging, sequencing, and routing. Such high level tools relieve courseware authors from learning programming languages, and shorten the courseware development time.

6.2 Courseware Development Philosophy

PLATO advocates the approach to designing courseware by faculty members themselves directly. Also PLATO preguesses the learners' response and its interactive session is mainly dialogue-based. Thus in PLATO no underlying instructional strategy can be built in. Each author provides his/her own pedagogical features. Courseware authors may reinvent the wheel.

In contrast, TICCIT advocates the approach of courseware design team, which consists of three faculty members, a programmer, and an instructional designer [Dea79] [Ald78]. Also specific learning strategies are used in the preparation of TICCIT materials. Specifically, learners can choose their learning strategy.

6.3 Application of Artificial Intelligence Techniques

Individualized learning requires computers to control the learning strategy. One easier way to do it is for the courseware authors to help the student to become a good learning strategist. Students can choose the difficulty of rule, example and practice combinations involved. The other way is using AI techniques [Ric83] to monitor students' response and then issue proper level of tutorial, example, and question to the students automatically.

AI techniques can also be used to train computers to accommodate more flexible answers (unlike arithmetic), and to devise a proof checker to determine lesson mastery.

Japanese 5th generation computer project and the US's booming AI technology [Bar81] have impact on the progress of learning algorithms, knowledge representation, and problem solving. Because of the requirement of large memory and fast processing power in such applications, there is no commercially viable products in CAI area. A detailed survey of AI applications in CAI can be found in [Enn83].

6.4 Computer Systems for CAI Courseware

Some courseware is designed to run on a timesharing mainframe computer and learners interacts with the system through terminals. For example, PLATO is built around CDC Cyber computer and Magnovax plasma terminals to support complex courseware having graphics and voice capabilities.

Another approach is to run courseware on the low cost microcomputers. MECC (Minnesota Educational Computing Consortium) develops courseware on Apple II microcomputers

[MEC79].

6.5 Courseware for Teaching Foreign Languages

One of the important applications of CAI is teaching foreign languages. A few of courseware for teaching foreign languages has been produced since 1960's. For examples, Stanford University's Russian, Chinese, and Armenian course [Sup75], U. of Surrey's German course [Ahm81], and PLATO's French, Hebrew, Hindi, Italian, Korean, Latin, Persian, Spanish, Swedish [Lym77] etc.

Besides computers, video tapes were often used in teaching foreign languages. Since language instruction requires more interaction, video disks emerge as a new substitute supporting interactive access. A video disk can store large amount of frames (54000 each side) longer than other media at an affordable price ($100 per disk). Video disks are typically used in the following way. One audio track is for foreign language vocabulary and another track for corresponding English vocabulary. As the instructor on screen says a word, the student imitates him/her, adjusting the film speed to slow motion or freeze frame if necessary.

6.6 Chinese Character Input/Output

Chinese characters are basically graphics; so typing input and output Chinese character becomes complex. On the other hand, each Chinese character has richer meaning and has only one vowel. To express the same thing, the number of keystrokes of typing Chinese characters (not using big keyboard) may be about the same as that typing in English. However, the training time needed for typing Chinese characters is longer than typing alphabetics.

Voice input is much attractive for Chinese since each Chinese character has only one syllable. However, the technology of voice recognition has not yet left the laboratories.

The Chinese character output device is usually a graphics display (raster scan, vector display or storage tube) or printer (dot-matrix, thermo, laser, or ink inject). The available Chinese character input devices are primarily keyboards. There are three types of keyboards, and four input methods. (1) Regular alphanumeric keyboard which can be also used to input English. There are three ways to input Chinese characters using such keyboard. The first method is typing the numeric code representing the character, which is often used in the telegraph application. The second method is to let each key represent a radical

(subcomponent of a character). Then type in a character by composing its component radicals based on the following building rule: top-to-bottom, left-to-right, and outer-to-inner. Two systems using this method are Multitech in Taiwan and Eastern Computers in Virginia. The third input method is based on transliteration [Wan83]. The characters with the same phonetic spelling and tone are then determined by the users. One product employing this method is carried by Microstar in D. C.. (2) Augmented alphanumeric keyboard, which can also be used to input English alphabets. Usually the input method is based on radical combinations. Since more keys are available to represent more subcomponents, the needed keystroke number can be reduced. Transtech in Massachusetts manufactures such product. (3) Big keyboard. Each frequently used character has a corresponding key in the big keyboard. The input method for such characters are input by typing once. Less frequently used characters can be constructed using some convention. Wang Lab. in Taiwan produces such product.

6.7 New Commercial CAI systems

As CAI research becomes mature, lately many CAI systems have been developed and marketed, not as a research means, but specifically for commercial use. Among them are:

6.7.1 PCIS of IBM In September 1984, IBM announced [IBM84] that it will market a training system, called Personal Computer Instruction System (PCIS). Also in December 1984, Raster Technology announced a product Teacher-Turned-Author [Ras84]. Both systems provide color graphical and text capabilities, and were claimed to be user-friendly to allow those with no programming skills to create, present and administer CAI on IBM PC.

A deficiency with these systems is that no voice capability is provided. As a result, they are inadequate for language instruction.

6.7.2 IVIS of DEC In January 1984, Digital Equipment Corporation (DEC) announced its Interactive Video Information System (IVIS) [DEC84]. Based on optical disk and DEC's own micro-computers, IVIS provides interactive instructions via color monitors with voice capability.

The targeted market seems to be for popular courses, such as mechanic training, X-ray photo examination, etc. The course generation cost is high, since the course-generation system of IVIS requires a VAX-750 or larger mini-computer. Also, the creation of optical disk is too expensive for relatively less-popular courses such as Chinese language instruction.

6.7.3 PLATO of CDC Control Data Corporation (CDC) is among the earliest player in the CAI field. It has been marketing timesharing CAI system PLATO for more than five years. PLATO [CDC83] is running on CDC Cyber mainframe computers and users access the systems through CDC's proprietary plasma terminals. Although PLATO has more than 6,000 hours of completed instructional material, its software and hardware structure is still based on the technology of 1976. Furthermore, the equipment lease fee (with four terminals) is over $30,000 per year, which does not include data line cost.

7. Bibliographies

[Ahm81] Ahmad, K. and Rogers, M., "Development of Teaching Packages for Undergraduate Students of German," Computer Simulation in University Teaching, North-Holland, 1981.

[Ald78] Alderman, D. L., "Evaluation of the TICCIT Computer-Assisted Instruction System in the Community College," Educational Test Service, Princeton N. J., 1978.

[Avn76] Avner, E., "Summary of TUTOR Commands and System Variables," Computer-based Research Lab., U. of Illinois at Urbana, 1976.

[Bar81] Barr, A. and Feigenbaum, E., The Handbook of Artificial Intelligence, Vol. II, 1981.

[Boe78] Boehm, B. W., Characteristics of Software Quality, North-Holland, 1978.

[Bor81] Bork, A., Learning with Computers, Digital Press, 1981.

[Cal79] The ABC's of CAI, Fourth Edition. California State Univ., Fresno, 1979.

[CDC83] CDC PLATO, Product Description. Control Data Corporation, Minnesota, 1983.

[Cha80] Chambers, J. A. and Sprecher, J. W., "Computer Assisted Instruction: Current Trends and Critical Issues," Communications of ACM, June 1980.

[Chi77] Modern Chinese Readers, Commercial Press, 1977.

[CON79] CONDUIT. Catalog of CONDUIT reviewed and tested materials, Pipeline, 1979.

[Dea79] Dean, P. M., "Computer Assisted Instruction Authoring Systems," Educational Technology, 1979.

[Dec84] Digital Equipment Corporation, Maynard, Massachusetts, Interactive Video Information System (IVIS), announced January 1984.

[Dia84] Dialogic Corporation. "Manual for DIALOG Voice System," 1984.

[Dij76] Dijkstra, E. W., "A Discipline of Programming," Prentice-Hall 1976.

[Ele84] "Best Ten," Electronic Products, April 20, 1984, p.31.

[Enn83] Ennals, E., "Artificial Intelligence," Computer Based Learning, Rushby, N. J. (ed), Programon Infotech Ltd. Berkshire, England.

[Gla84] Glatzer, H., "Opening the Curtains on Windowing," Software News, July 1984.

[GKS84] "American National Standard Graphical Kernel System," Technical Committee X3H3 -- Computer Graphics, Special GKS Issue, Computer Graphics, February 1984.

[Gss84] Graphical Software Systems, Inc., "Manuals for GSS Drivers and GSS Tools," 1984.

[Hin84] Hinden, H. J., "Special Report on Graphics Technology," Computer Design, May 1984, p.163 - p.234. 7 articles.

[IBM84] International Business Machinery Inc., Product Center products: Personal Computer Instruction System (PCIS), announced 1984.

[IEE83] IEEE Computer Society Workshop on Software Engineering Technology Transfer, Proceedings, April 1983.

[Lym77] Lyman, E. R., "PLATO Curricular Materials, No. 6," CERL Report, Nov. 1977.

[Mar83] Mark Rawlins, "VDI - the New Graphics Alchemy," Electronic Products, August 1983.

[Mcl84] McLeod, J., "Special Report on Graphics," Systems and Software, August 1984, p.82 - p.95.

[Mcl84a] McLeod, J., "Optical Disks: Mass Storage of Information," Systems and Software, December 1984, p.102-p.115.

[MEC79] Microcomputer Report. Minnesota Educational Computing Consortium, Instructional Services Division, Lauderal Minnesota, July 1979.

[Pio82] Pioneer Video Inc., "Scheduling, Authoring, and Project Management Guide," 1982.

[Pio83] Pioneer Video Inc., "Manuals for Laserdisc Player," 1983.

[Ras84] Raster Technology, Inc., Product "Teacher Turned Author!" PC World, December 1984, p.259.

[Raw83] Rawlins, M., "VDI - the New Graphics Alchemy," Electronics Products, August 1983.

[Ric83] Rich, E., Artificial Intelligence, McGraw-Hill, 1983.

[Rot84] Rothchild, E. S., "Optical Memory: Data Storage By Laser," BYTE, October 1984, p.215 - p.226.

[Rus83] Rushby, N. J. (ed), Computer Based Learing - State of the Art Report, Programon Infotech Ltd., Berkshire, England, 1983.

[Schu84] Schuyler, C., "Interactive Video Information Systems," Computer World, October 29, 1984, p. ID13 - p. ID16.

[Spe78] "A Second Chance for Computer-Aided Instruction," IEEE Spectrum, Aug. 1978.

[Sup75] Suppe, P., Smith, R. and Beard, M., "University Level Computer Assisted Instruction at Stanford," Technical Report Number 265, Stanford U., IMSS 1975.

[Wag84] Waggoner, C. N., "GKS-based Graphics Software Adapts to Changing Technologies," EDN, May 1984, p.31.

[Wan83] Wan, S., Saitou, H. and Mori K., "The Feasibility of Chinese Word Processing Using the Pinyin-Hanzi Conversion Method," Proc. of ICTP (International Conference on Text Processing with a Large Character Set), 1983.

[Wir74] Wirth, N., "On the Design of Programming Languages," Proc. IFIP Congress, 1974, p.386-p.392.

[Yaj83] Yajima, K. and Hirashima, M., "Consideration for the Development of Arabic Word Processor," Proc. of ICTP, 1983.

APPENDIX A

User Interface Specification for the MMAS Authoring System

Introduction

This section states the specification of the user * interface for the MMAS authoring system. It describes, from a user's point of view, how the authoring system can be used to make a courseware.

The purpose of this Appendix is to describe all the options available, so that this can serve as a specification for the next phase of software implementation. If the reader simply wants to get some understanding of the user interface, he(she) can skip most of the context and read only those examples.

User Environment

The user environment consists of: a color graphic monitor, a microphone, a speaker, a keyboard, and a mouse. # The microphone, keyboard, and mouse @ are the input device, from which the user enter his (her) instructions to the system. The mouse is a device that the user uses to point a place on the monitor, select instructions, and execute instructions. The speaker and the monitor are output devices, from which the system output data for user to hear and view. The color monitor is where the user see:

---

* In the context of this Appendix, the user means the person who uses the authoring system to create or compose courseware. In other words, the user is the author of a courseware. Therefore, the words "author" and "user" will be used interchangeably in the text.

For more detailed discussion of hardware, the reader is referred to Section 2.3.

@ A mouse device is optional in the MMAS authoring system. In other parts of this proposal, keyboard is assumed. The movement of a mouse can be simulated by a keyboard.

1. what was typed in, and

2. what the system is recording from the microphone.

Without doubt, the monitor screen presents the most information, and therefore will be the focus of our presentation.

Terminologies

This section defines a few terminologies that are used extensively in this Appendix. For common terminologies, the reader is referred to Appendix C, which describes a model of CAI applications and some definitions of terminologies.

A courseware (henceforth referred simply as a course) consists of many frames of educational materials. A frame here is a scene, including both screen and voice, that will be played at the delivering system and will be received and reacted by the student.

Summary

The user interface can be summarized as three scenes, each of which will be discussed in the next three sections. The first section "Course Flow Organizer" describes a capability of the MMAS, which provides the user with a mechanism to organize his (her) course. What the author sees at the screen is for the convenience of the author, and will not be seen by the student at the delivering system, where the course is to be played. The second section "Course Frame Editor" is about the MMAS capability that provides the user with tools to construct and edit each frame. The last section "Course Rehearser" describes the capability of "play back" part of the courseware being constructed.

Course Flow Organizer

This tool allows the author to organize his (her) course before and during the construction of the course.

An interactive course is different from other one-way course in that it receives responses from the student. Consequently, it is desirable to provide the capability for the author to specify different response for various student answers. For example, the author may want to give a detailed explanation to a student when he (she) incorrectly answers a quiz.

Using the organizer, the author creates a flow-chart of the course. The flow-chart indicates the sequences that a

course will be presented to the student as well as conditions the course be changed with respect to the response of student.

A typical course flow organizer screen is composed of commands and course structure. On top of the screen are a list of commands: EXIT, VAR, LIB, EDIT, and REHEARSE.

EDIT provides a table of capabilities to create, modify, and erase the course flow chart. Specifically:

1. Create-block
   allows the creation of either a frame or a set of frames. When a block is created, it is automatically assigned with a unique identifier.

2. Link
   allows the linking of two blocks. One is the starting and the other is the ending block. Each block can have one inward link and one outward link. When two blocks are not within the distance of a screen, the block identifier can be used to link them. The possible branches may be allowed, however, by the following feature.

3. Action
   allows the creation of a conditional branch in the course. The author can specify any Boolean expression.

4. Erase
   erases the component(s) on the screen. Since the objects BLOCK, LINK, and ACTION do not overlap, moving the cursor to the object and enacting the Erase operator automatically remove the object from the screen.

5. Moving-Up (or Moving-Down)
   rolls the flow-chart up (or down), the screen can cover a part above (or below) the current position.

To provide information for the Action node to act on, the author can specify statistics bins, which are listed in a special window as the VAR. The use of the bins are numerous. For example, the author may specify a statistics bin to keep track of the score of the student. Every time the user answers correctly a question, the corresponding action node adds a weight to the bin. At certain action node in the course, the author may want to compare the action node with a fixed score. If the score is higher or lower than a certain score, the student may be allowed to

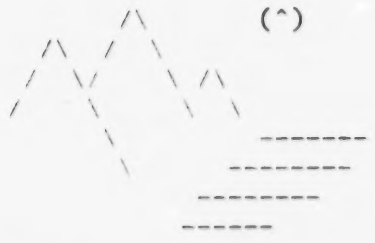
proceed to the next more advanced session.

There are also a list of library routines to allow the author to incorporate into the action node. One example of the library routines is the random number generator. Using a random number to address a word in a dictionary, a course can generate literally infinite sessions.

Course Frame Editor

This section describes features that allow the author to create the exact screen image and voice sound to be seen/heard by the student.

In this system, each course is composed by many scenes. Each course is known to the student with a topic, such as "social conversation", "department store", etc. A scene is basically a screen image with potentially many voice sequences and updates to the scene. The features provide scenes to be overlapped with each other as well as be displayed individually.
An example can illustrate this well: Suppose the author wants to teach a few Chinese words about country scenery, hills, rivers, etc. He (she) may choose to use the editor features in this section to create a simple screen image with lines to indicate hills, rivers, and clouds. (To simplify the courseware volume, this system does not provide the features to create arbitrary graphic objects. Simple objects, however, can be approximated with lines and shades. The following figure illustrates one with all broken lines.



With this frame as background, he wants to make instruction sequence to teach Chinese characters corresponding to mountain, river, and cloud.** As the first "update", the

author moves the cursor to the place on the screen where drawings are made for the mountain, place the Chinese character "hill", and place a line of English text at the bottom of the screen: "This is Chinese Character 'Hill'. Hear the pronunciation of it, and then enter RETURN key." This update does not need completely redrawing the screen image. It instead augments, or overwrites a part of the graph.

At the second "update", triggered by the key input from the student, the speaker pronounces the Chinese character of hill (like tha'ine). The spelling of the Chinese "hill" appears next to the character. In the mean time, the English text appeared at the bottom of the screen is replaced by another message: "Enter RETURN to proceed, enter ? to hear again."

In case the student enters the RETURN key, the third "update" is projected to the screen. It places the Chinese character "river" to the place where river is drawing on the screen. Similar English text appears on the screen, and so on.

Note that each scene keeps the same screen image with hills, rivers, and clouds. Only a part of the screen is changed. In this way the author can compose one frame and use it to teach much of the material he (she) wants to cover without the need to draw many screen images. This is because we realize that an author is not a cartoonist, and many authors may be more interested in constructing the course than composing graphics.

Course Rehearser

During the preparation of a course, this system provides the author with the capability to play back the course as feedback. This capability allows the authoring system to become temporarily a student system. The screen shows what the student is to see, what the student is to hear, and what the system reacts to the input from the student according to the courseware. The only difference from a true student system is the commands listed at the top of the screen.

** The facility to prepare this document allows only characters as output. In the real MMAS system, the graphical capability would create a graph of much better quality.

which enable the author to get control of the system and enter either the Organizer mode or the Editor mode described in the last two sections.

The rehearser can be started at any frame in the organizer mode. It is started when the author moves the cursor to the Command REHEARSAL and releases at any frame. The screen displays that frame and voice, if any. When the answer is entered to the system, it reacts according to the flow chart, which describes how the courseware is constructed, controlled in the organizer.

The Rehearsal mode provides the author with a distinct set of commands to control the section. They include:

- BACK
- GO
- PAUSE
- REPLAY this frame
- Single frame, where this frame is displayed until a GO command is triggered.
- EXIT
- HELP
- To organizer mode

Appendix B Data Base Structure for Authoring System

This Appendix gives a full definition of internal data base structure for the node_file discussed in Section 2.3.3.

To provide full syntax, a C-language format is used. For readers with programming language knowledge, this specification would describe the design in a more rigorous sense than a pure English description as in the text of this report. Note that this does not imply that the final implementation must be implemented in C, although C is a nice algorithmic, high level language like Pascal and Fortran. The structure here is rather to provide the information necessary for the courseware. An equivalent structure for the Fortran or Pascal can easily be translated from the structures outlined here.

```
#define UINT16  unsigned short
#define SINT16  short

struct controlnode {
        struct n_control {
                UINT16 nc_order;        /* the order in which graph,
                                           text, voice be presented */
                UINT16 nc_funct;        /* pointer to the function string */
                SINT16 nc_1next;        /* 1st choice node to goto */
                SINT16 nc_2next;        /* 2nd choice node to goto */
                SINT16 nc_1from;        /* 1st node where control came from *
                SINT16 nc_2from;        /* 2nd node where control came from *
        } n_control;
        struct n_graph {
                UINT16 ng_status;       /* UPDATE, NEWSCREEN, DONTCARE */
                UINT16 ng_pointer;      /* the offset to the graph VDM frame
        } n_graph;

        struct n_text {
                UINT16 nt_attribute;
                UINT16 nt_pointer;
        } n_text;

        struct n_voice {
                UINT16 nv_attribute;    /* delay time, speed, */
                UINT16 nv_pointer;      /* the offset to the voice file */
        } n_voice;

};
```

Each of the data structure "controlnode" corresponds to one "SCENE" or "UPDATE" as defined in Appendix A. Using user-oriented description, every time the author creates a SCENE or initializes an UPDATE for a SCENE, one of the data structure is allocated for that entity. The control of the course, the conditions under which the course advances from one frame to another frame, is described in the sub-structure "n_control". Each node may contain a graph part, which corresponds to the sub-structure "n_graph", a voice part, which corresponds to "n_voice", and a text part "n_text". If a part of these components is not necessary, (for instance, the author may decide there need no voice for one scene.) the the corresponding "attribute" field will contain 0 (zero).

In the sub-structure n_control, the field "nc_order" indicates the order the graph, text, and voice should be presented to the student. For instance, the value 123 in the field means that the graph should be output to the screen first, then output the text portion, and finally, after all graphic information is displayed, output the voice portion. This is also the default for all course frames.

The field "nc_func" points to a character file of action commands, the action file, which corresponds to the actions that an author can specify. An example of the action is "if (SCORE > TIME + 60) exit; SCORE=SCORE+1,goto 1."
It says that for the two bins "SCORE" and "TIME", if the SCORE is higher than the sum of TIME and sixty, the course if completed and exit the course. If the score is not higher, SCORE is increased by one and the next scene, as indicated in nc_1next field, is to be displayed on the screen.

The fields "nc_1from" and "nc_2from" represent the frame addresses in the node file to record where control may come from.

Graphical information for each frame is described in sub-structure "n_graph". The field "ng_pointer" is the address to the graphic file "graphfile" (refer to File Description in this Appendix) , where all graphic information is stored.

APPENDIX C GLOSSARY

In order to describe the design, we shall first introduce some terminologies that are used widely in the rest of this proposal. Figure C-1 shows a simple model of CAI that we perceive.

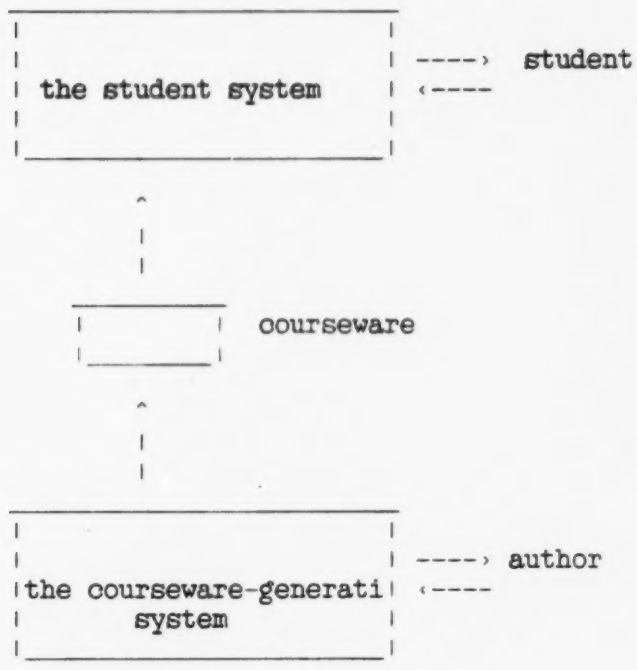


Figure C-1 A Simple CAI Model

In the figure, the CAI process is recognized as two stages: authoring and delivering. In the stage of authoring, the courseware is developed and generated on a so-called courseware-generation-system. The finished courseware is moved to the student-system (or delivering-system) via storage media like tape and disk, or even via connected cable or telephone link. As the name suggests, the student system interacts directly with students via interfaces like screen, speaker, keyboard, keypad, and other special

devices, if any.

With this simple model in mind, following are a list of commonly used terminologies:

1. authoring system

   The computer system equipped special software or hardware to interact with an author to facilitate the creation of courseware.

2. courseware

   Courseware is computer generated or controlled instruction materials with which learners can interact through audio, video, touching, or typing devices.

3. courseware-generation system

   This is the same as "authoring system."

4. courseware-delivering system

   The computer system equipped special software or hardware to interact with a student, based on the courseware, to facilitate the learning process of the student.